# MOBILEMEM: EVALUATING LONG-HORIZON MEMORY FOR LANGUAGE AGENTS IN REAL-WORLD MOBILE ENVIRONMENTS

**Xinle Deng**[1], **Yida Xue**[1], **Yijun Chen**[1], **Mingjun Mao**[1], **Ruobin Zhong**[1], **Buqiang Xu**[1], **Jizhan Fang**[1], **Haoming Xu**[1], **Tingwei Wu**[1], **Yajing Xu**[1], **Shumin Deng**[2], **Haofen Wang**[3], **Huajun Chen**[1], **Ningyu Zhang**[1*]
[1] Zhejiang University [2] National University of Singapore [3] Tongji University
{dengxinle, zhangningyu}@zju.edu.cn

## ABSTRACT

Long-term memory is widely regarded as a key enabler of personalization for Large Language Model (LLM) agents, yet existing benchmarks almost exclusively model users through human–assistant dialogues, implicitly assuming that user preferences can be fully inferred from conversational signals alone. However, an effective personalized memory system should not be limited to conversations, but should be learned from continuous observations of diverse user behaviors, a setting that remains largely unexplored due to the lack of appropriate benchmarks. To this end, we introduce **MobileMem**, a benchmark for evaluating personalized long-term memory in realistic environments, using mobile usage as a representative and challenging testbed. MobileMem is constructed from real user trajectories, where human-assistant dialogues are naturally interleaved with interactions across multiple mobile applications. To enable coherent long-horizon evaluation from fragmented sessions, we further propose KEME, a knowledge-guided experience synthesis framework that integrates temporally dispersed interactions into consistent lifelong trajectories. Each trajectory is paired with long-horizon question–answer pairs that require memory systems to organize, retrieve, and integrate information across sources and time. Evaluations on MobileMem expose previously overlooked limitations of existing memory systems, revealing a significant gap between current benchmarks and real-world deployment demands.

## 1 INTRODUCTION

Large Language Model (LLM) agents are increasingly envisioned as long-term personal assistants capable of continuous learning and adaptation. To achieve true personalization, these agents must possess the ability to accumulate, integrate, and reason over user-specific information across extended time horizons spanning days, weeks, or even months Jiang et al. (2024); Chen et al. (2024); Liu et al. (2025b); Cai et al. (2025); Sun et al. (2025a). Recent research demonstrates that incorporating long-term memory systems can substantially enhance the personalization, behavioral consistency, and long-horizon task performance of LLM agents Chhikara et al. (2025); Xu et al. (2025); Rasmussen et al. (2025); Li et al. (2025); Kang et al. (2025); Wang & Chen (2025); Fang et al. (2025a); Wang et al. (2025a).

However, the rapid development of memory systems has outpaced our ability to rigorously evaluate them under realistic usage conditions. Most existing benchmarks implicitly operationalize memory through a dialogue-centric lens, assuming that a user's persona and preferences can be fully inferred from human-assistant conversations alone Maharana et al. (2024); Wu et al. (2025a); Tan et al. (2025); Hu et al. (2025a); Chen et al. (2025). We argue that an effective personalized memory system which we term multi-source memory systems should not be limited to explicit dialogues. Instead, it must learn from continuous observations of diverse user behaviors. As illustrated in Figure 1, this paradigm shift offers two critical advantages: **comprehensive context completion** and

* Corresponding author.

**holistic preference perception**. First, observations provide the "missing state" of a user that dialogue often omits. In Figure 1, while a dialogue-only system might suggest "Roast Beef" based on general Sunday habits, it fails to recognize the user's immediate physiological state. In contrast, by observing a note about a recent 42km ride, a multi-source system can complete the context of "intense physical exhaustion" and suggest a more appropriate option, "Acai Bowl". Second, many latent preferences are embedded solely in non-conversational interactions. For instance, the user's preference for "nuts after exercise" is recorded in a private note but never mentioned in past dialogues. Existing benchmarks, even those focusing on standalone streams of visual observations Chandrasegaran et al. (2024); Yang et al. (2025b); Wang & Chen (2025); Long et al. (2025); Jiang et al. (2025b), largely overlook the temporal and cross-source heterogeneity inherent in such fragmented mobile interactions, leading to a significant gap between laboratory evaluation and practical agent deployment.

Motivated by these insights, we introduce **MobileMem**, a benchmark designed to evaluate personalized long-term memory within realistic, multi-source mobile environments. Unlike prior work, MobileMem models the user experience as a unified stream of heterogeneous messages originating from both human-AI dialogues and diverse mobile applications. This captures the fragmented nature of real-world personal memory across various formats and semantics. To ensure realism while maintaining privacy, MobileMem employs a realism-guided data construction paradigm[1]. We ground user personas in interviews with real individuals and synthesize application interaction traces that follow realistic schemas and temporal distributions, ensuring all sensitive attributes are strictly anonymized.

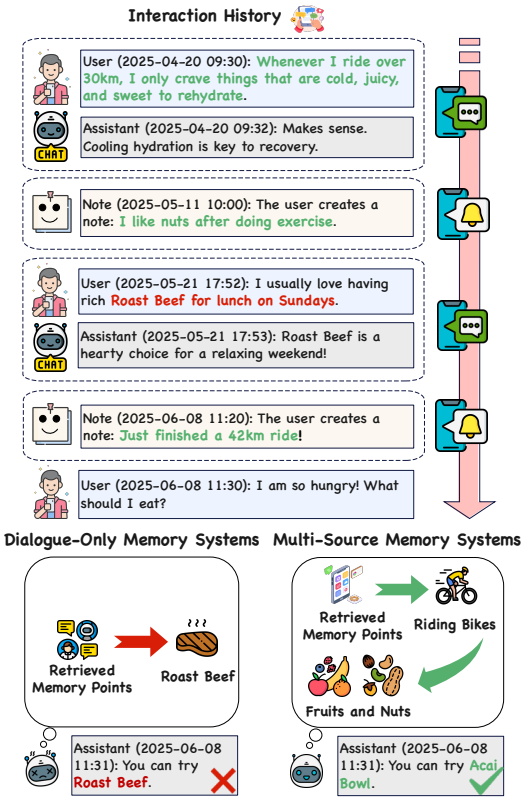


Figure 1: Comparison between *dialogue-only memory systems* and *multi-source memory systems* in a mobile usage scenario.

Constructing coherent, long-horizon trajectories from such fragmented and heterogeneous sources poses a significant technical challenge, particularly when attempting to synthesize and interleave human-assistant dialogues naturally with background application activities. To address this, we propose Knowledge-guided Experience synthesis for evolving MEmory (KEME), a hierarchical framework that integrates dispersed user-app sessions and synthesized user-assistant dialogues into consistent lifelong trajectories. Using KEME, we produce a dataset spanning 448 days, comprising 1,269 dialogue turns and 1,336 questions. **MobileMem is released under the MIT License, and all collected data are anonymized and reviewed by the Institutional Review Board (IRB)[2] to ensure privacy protection**.

MobileMem adopts an end-to-end evaluation protocol where systems are required to answer long-horizon question-answer pairs after processing an entire interaction trajectory message-by-message. This setup requires the memory system to autonomously organize, retrieve, and integrate information across disparate sources and vast temporal gaps. Our empirical evaluations expose clear limitations in existing memory baselines: current methods struggle significantly with heterogeneous source integration and observation-based reasoning. These findings highlight a substantial gap between current benchmarking practices and the demands of real-world deployment, positioning MobileMem as a vital testbed for the next generation of personalized memory systems.

[1]All data are collected under signed agreements, with users consenting to public release, and undergo rigorous de-identification to ensure privacy.

[2]The data are sourced from a well-known mobile device manufacturer.

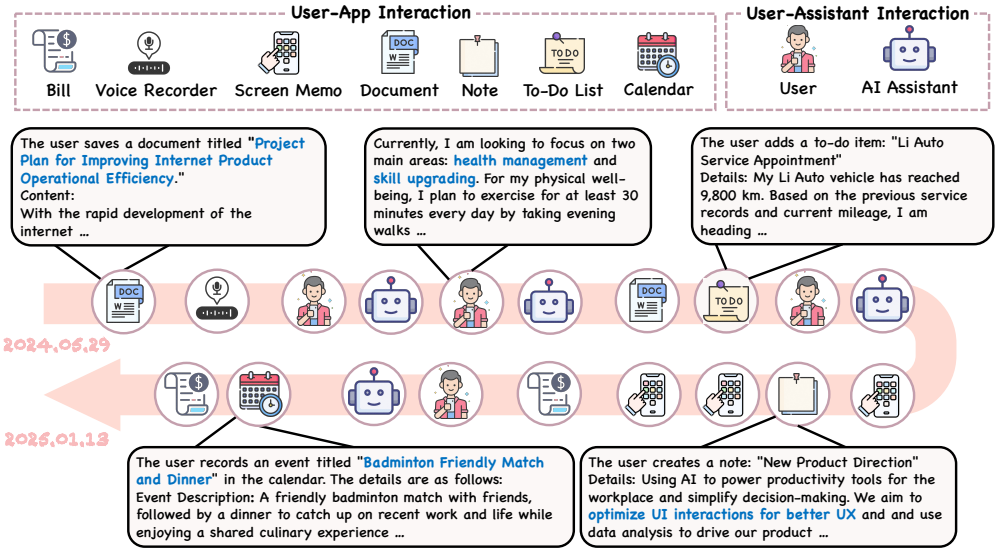


Figure 2: **A user's trajectory in MobileMem.** It comprises 9 distinct sources categorized into two primary interaction types: *User-App Interaction* and *User-Assistant Interaction*. The trajectory contains heterogeneous content and maintains high logical consistency, effectively reflecting the user's personal profile.

## 2 Construction Pipeline for MobileMem

### 2.1 Design Principle

MobileMem is designed to evaluate memory systems in real-world mobile environments. To achieve this goal, it follows these principles below:

**Heterogeneity and Multi-Source.** Real-world mobile environments include not only human-assistant dialogues but also a complex ecosystem of third-party applications. User interactions across these apps are inherently heterogeneous in both format and semantics. MobileMem is designed to reflect this reality by requiring memory systems to process real-time message streams originating from diverse application sources.

**Observation and Participation-Based Interactions.** In real-world mobile scenarios, users spend the majority of their time interacting with various applications rather than directly engaging with an AI assistant. To ensure a seamless user experience, an assistant equipped with a long-term memory system must operate in a non-intrusive manner, learning from the user's activities through passive observation rather than interrupting their workflow. Consequently, trajectories in MobileMem are designed to interleave participation-based interactions (explicit human–assistant dialogues) with observation-based ones (passive monitoring of user–application interactions), reflecting the true nature of an ambient intelligent assistant operating continuously in the background.

**Realism-Guided Synthetic Data.** In real-world mobile environments, user interactions naturally contain a wide range of sensitive personal information Pentina et al. (2016). Collecting raw, unprocessed logs from multiple apps poses significant privacy risks, including potential re-identification and the exposure of confidential personal attributes Su et al. (2017); Tu et al. (2018). To mitigate these concerns, MobileMem adopts a realism-guided construction paradigm.

### 2.2 Problem Formulation

Each instance in MobileMem is formalized as a tuple $(\mathcal{T}, \mathcal{Q})$, where $\mathcal{T}$ represents the user's interaction trajectory, and $\mathcal{Q} = \{(q_j, \mathcal{A}_j)\}_{j=1}^{|\mathcal{Q}|}$ denotes the set of question-answer pairs associated with the

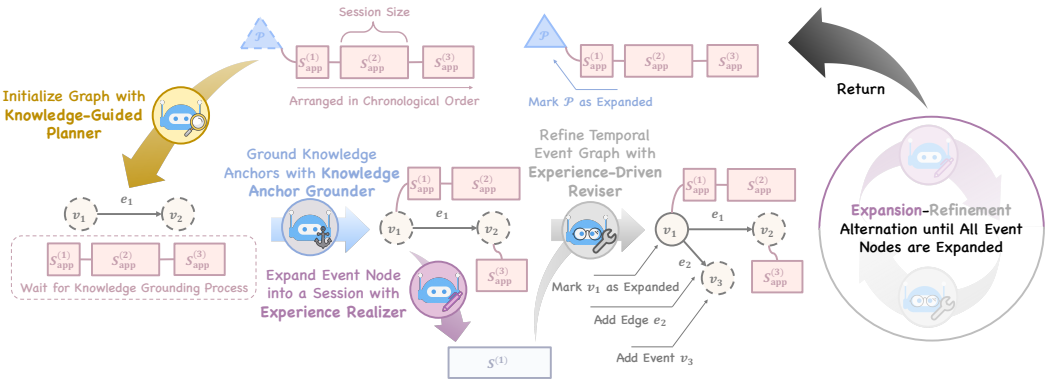


Figure 3: **An illustrative example of KEME.** For simplicity, the example shows a maximum expansion depth of 1, and persona evolution operations are omitted. The process begins as the knowledge-guided planner initializes a temporal event graph with nodes $v_1, v_2$ and edge $e_1$ based on the initial user persona $\mathcal{P}$. The knowledge anchor grounder then assigns inherited external sessions $S_{\text{app}}$ to compatible event nodes within this initial graph. Following a topological order, the experience realizer expands an event node into a concrete session $S^{(1)}$ by merging grounded anchors such as $S^{(1)}_{\text{app}}$ and $S^{(2)}_{\text{app}}$. Based on this newly synthesized experience, the experience-driven reviser dynamically updates the remaining graph by adding a new event $v_3$ and edge $e_2$ to maintain structural richness. This *expansion-refinement alternation* cycle continues until all event nodes are expanded, resulting in a coherent, long-horizon trajectory.

trajectory. Here, $\mathcal{A}_j = \{a_k^{(j)}\}_{k=1}^{|\mathcal{A}_j|}$ is the set of golden answers for question $q_j$. In MobileMem, the user trajectory is modeled as *a stream of heterogeneous messages*, which contains not only direct human-assistant dialogues but also interaction logs with external applications. Typically, this continuous stream is segmented into discrete sessions $\mathcal{T} = \{\mathcal{S}_j\}_{j=1}^{|\mathcal{T}|}$ based on a predefined segmentation criteria Wang et al. (2022); Xu et al. (2022); Wang et al. (2025d), where $\mathcal{S}_j = \{m_k^{(j)}\}_{k=1}^{|\mathcal{S}_j|}$ represents the $j$-th session containing a sequence of messages from the user, the AI assistant, or applications.

For notational simplicity, we flatten the trajectory into a single sequence $\tau = \text{Flatten}(\mathcal{T}) = \{m_i\}_{i=1}^{|\tau|}$. Let $\mathcal{M}_0$ denote the initial memory state. At step $t$ ($t \geq 1$), the system receives message $m_t$ and updates the memory state to $\mathcal{M}_t$ via

$$\mathcal{M}_t = \text{Update}(\mathcal{M}_{t-1}, m_t, T(m_t)), \tag{1}$$

where $\text{Update}(\cdot)$ denotes the memory update mechanism and $T(\cdot)$ denotes the timestamp function. We adopt an end-to-end strategy to evaluate the memory system, following previous memory benchmarks Wu et al. (2025a); Chen et al. (2025). Specifically, For each question-answer pair $(q, \mathcal{A}) \in \mathcal{Q}$, the memory system generates an answer $\hat{a}$ based on the final memory state $\mathcal{M}_{|\tau|}$, which integrates the information from the entire trajectory $\tau$. The correctness is assessed against the reference set $\mathcal{A}$ via LLM-as-a-Judge Gu et al. (2024).

### 2.3 Sources

As detailed in Figure 2, we consider two major categories of message sources. The first is *user-app interaction*, where messages originate from user interactions with various external mobile applications. The second category is *user-assistant interaction*, consisting of direct conversations between the user and an on-device AI assistant.

Regarding user-app interactions, we incorporate data from 7 apps (see Appendix A.1). Collectively, they capture a broad spectrum of activities, ranging from information browsing (*Screen Memo*) and professional documentation (*Document*) to personal scheduling (*Calendar*) and financial management (*Bill*). Notably, the data collected from these apps exhibit significant heterogeneity in both structural format and semantic depth. For example, a document may contain a structured, formal project plan regarding operational efficiency, whereas a to-do list might consist of a brief, informal task like a service appointment.

### 2.4 PERSONA CONSTRUCTION

To ensure diversity and realism in user behavior, we construct personas based on real individuals rather than purely synthetic profiles. We **recruit volunteers with diverse demographic backgrounds**, spanning different occupations, cities, age groups, and personal interests. We define a unified persona schema (see Figure 5) that includes attributes such as basic demographics, professional background, lifestyle patterns, and personal preferences. All persona profiles are collected through interviews conducted by the data source's staff.

For privacy protection, sensitive attributes are anonymized and adjusted while preserving their behavioral roles. As a result, the constructed personas remain realistic while not corresponding to any identifiable real individuals.

### 2.5 USER-APP INTERACTION SYNTHESIS

We collect interaction trajectories from volunteer smartphones over a one-month period. To strictly adhere to application service agreements and privacy standards, our data collection process is designed to record only app usage statistics and associated metadata, explicitly bypassing raw interaction content. The trajectories also incorporate user-centric context, including geographic location, Bluetooth connectivity, and network status. Following anonymization, we synthesize the specific semantic content of user-app interactions based on the collected trajectories.

To synthesize the semantic content for memory system evaluation, we must first define the perception interface between applications and memory systems. Unlike existing Graphical User Interface (GUI) agents typically rely on screen-transition sequences Tang et al. (2025), we argue that a mature on-device memory system should operate on a safer, low-redundancy interface. Consequently, we simulate a protocol-based app-mediated perception (see the related discussion in Appendix D.1) environment for our data generation of user-app interactions. Instead of synthesizing raw screenshots, we assume that each app actively transmits structured messages to the memory system via a specific protocol, deciding which interactions are meaningful and secure to record. For example, in our simulation, a voice memo app sends transcribed text via a lightweight speech recognition model, while a to-do list uses rule-based triggers to report the creation or completion of tasks. Guided by this protocol-based design, we establish a perception protocol and predefined message template for every application in the dataset (see Appendix A.2). For each user, we define a topic distribution for every app based on his or her persona and trajectory. Beyond this, we assign specific temporal distributions for calendars and to-do lists, and word-count distributions for notes to reflect realistic usage habits. LLMs then generate specific interaction content through few-shot prompting, taking a trajectory, an app schema, and a persona profile as inputs. A notable exception is the synthesis of screen memos, which we construct using the Wikipedia API[3].

## 3 KEME: KNOWLEDGE-GUIDED EXPERIENCE SYNTHESIS FOR EVOLVING MEMORY

The synthesis method described in Section 2.5 provides realism-guided synthetic user-app session data. However, these sessions are fragmented and lack both long-term coherence and participation-based human–assistant interactions. To bridge this gap, we propose KEME, a knowledge-guided experience synthesis framework designed to construct evolving lifelong trajectories.

KEME treats the fragmented user-app sessions as *foundational knowledge anchors* that reflect what has already occurred and must be preserved. **Guided by user persona knowledge and temporal constraints, KEME hierarchically organizes these anchored sessions into a unified interaction stream, while progressively synthesizing human–assistant interactions that naturally emerge as the user's experiences unfold over time.**

Beyond trajectory synthesis, KEME integrates a *bottom-up synthesis algorithm* to produce high-quality question–answer pairs, enabling the systematic evaluation of memory systems under realistic and multi-source scenarios.

[3] https://wikipedia-api.readthedocs.io/en/latest/API.html#wikipedia

### 3.1 Formalization and Data Models

We formalize the synthesis process as a two-stage pipeline. In the first stage, given an initial user persona $\mathcal{P}_{\text{start}}$, a set of knowledge anchors $\mathcal{K}$ (instantiated as user-app sessions $\{\mathcal{S}_{\text{app}}^{(i)}\}_{i=1}^{N}$, where $N$ denotes the total number of available sessions), and a time horizon $[T_{\text{start}}, T_{\text{end}}]$, KEME generates a long-horizon, evolving trajectory $\mathcal{T}$. In the second stage, the framework constructs a set of question-answer pairs $\mathcal{Q}$ based on the synthesized trajectory $\mathcal{T}$, the final user persona $\mathcal{P}_{\text{end}}$, and an initial question type tool book $\mathcal{B}_{\text{start}}$.

To ensure robustness and facilitate automatic validation, KEME defines a suite of data models based on Pydantic[4]. The agents in our framework interact with these models through explicit tool calls, which allows for creation, modification and deletion of these data models. Appendix B.1 and B.2 provide full details.

### 3.2 Closed-Loop Trajectory Synthesis with Anchored Knowledge and Evolving Experience

The trajectory construction in KEME is organized as a *closed-loop process* centered on two principles: *top-down knowledge guidance* and *bottom-up experience evolution*. This design enables long-horizon coherence under anchored knowledge while progressively unfolding a person's evolving experiences over time. Operationally, KEME instantiates this loop with four specialized agents: a *Knowledge-Guided Planner* $\mathsf{A}_{\text{plan}}$, a *Knowledge Anchor Grounder* $\mathsf{A}_{\text{ground}}$, an *Experience Realizer* $\mathsf{A}_{\text{realize}}$, and an *Experience-Driven Reviser* $\mathsf{A}_{\text{revise}}$. Figure 3 shows an illustrative example of the closed-loop trajectory synthesis process. Algorithm 1 outlines the end-to-end trajectory construction pipeline.

**Top-Down Knowledge Guidance.** Trajectory synthesis starts from a root node $r$ that represents the initial persona $\mathcal{P}_{\text{start}}$, together with a time horizon $[T_{\text{start}}, T_{\text{end}}]$ and a set of knowledge anchors $\mathcal{K}^{(r)}$. Given $(\mathcal{P}_{\text{start}}, [T_{\text{start}}, T_{\text{end}}])$, the knowledge-guided planner $\mathsf{A}_{\text{plan}}$ constructs a root-level temporal event graph $\mathcal{G}^{(r)} = (\mathcal{V}^{(r)}, \mathcal{E}^{(r)})$ that partitions the horizon into a small set of coarse-grained life events. Each event is required to be temporally valid[5] and semantically compatible with the persona.

The planner then expands the graph in a recursive manner reflecting the hierarchical structure of human experiences. Each event node $n$ can be further expanded into either a finer-grained temporal sub-event graph $\mathcal{G}^{(n)}$ or a leaf-level session $\mathcal{S}^{(n)}$, thereby forming a multi-level decomposition from coarse life phases to concrete interactions. The expansion strategy depends on the hierarchy depth (other strategies are provided in Appendix B.3). If the current depth reaches $d_{\max}$[6], events are expanded into sessions, whereas at shallower depths events are expanded into sub-event graphs. At every recursion level, the parent node's temporal boundaries and semantic constraints (see related details in Appendix B.1.3), together with the current persona state, define the admissible space for descendant synthesis.

Knowledge anchors are integrated through a grounding step whenever the current parent node $u$ carries anchored sessions $\mathcal{K}^{(u)}$. Specifically, the knowledge anchor grounder $\mathsf{A}_{\text{ground}}$ assigns each anchored session in $\mathcal{K}^{(u)}$ to a compatible event node in the corresponding graph $\mathcal{G}^{(u)}$, where compatibility requires that the event time interval fully contains the session interval. When no compatible event exists, the agent revises $\mathcal{G}^{(u)}$ so that every anchor becomes groundable. For each event receiving anchors, the grounded content is summarized into an event-level *compatibility context*. This context is propagated to subsequent expansions as a non-contradiction constraint, ensuring anchored knowledge is preserved throughout the hierarchy and preventing synthesized human-assistant interactions from conflicting with the grounded sessions.

**Bottom-Up Experience Evolution.** Given a temporal event graph, the knowledge-guided planner $\mathsf{A}_{\text{plan}}$ expands event nodes by following a topological order. Importantly, each expansion produces

[4] https://github.com/pydantic/pydantic

[5] Its start and end timestamps must lie within the parent node's time horizon.

[6] We define the depth of the root node to be 0.

new concrete outcomes and constraints that are not fully available at planning time. Therefore, after expanding an event, the experience-driven reviser $\mathsf{A}_{\text{revise}}$ updates the remaining unexpanded parts of the corresponding graph. Concretely, $\mathsf{A}_{\text{revise}}$ may add, remove, or adjust future events and dependency edges, to reflect implications of the newly synthesized experience. These refinements aim to resolve emerging inconsistencies, prevent unnatural future transitions, and increase structural richness. This *expansion-refinement alternation* forms a bottom-up feedback loop that continually revises the graph as new experiences are synthesized.

At the leaf level, the experience realizer $\mathsf{A}_{\text{realize}}$ produces concrete sessions. If an event has grounded external sessions, the anchored content is directly adopted as the event outcome, or merged when multiple anchors are assigned, thereby preserving the immutability of anchored knowledge. Otherwise, $\mathsf{A}_{\text{realize}}$ synthesizes natural human-assistant dialogues that are consistent with the event context and all inherited constraints. Beyond generating interactions, sessions also drive *persona evolution*. We represent a persona as a set of dimension-wise attributes with explicit version histories, where each attribute version maintains message-level evidence links. Attributes are initially unrevealed in the sense that they have no linked evidence. During session synthesis, only user or system messages are allowed to be linked to persona attributes, turning them into evidence-grounded revealed states. Moreover, if the realized experience implies genuine state changes, such as updated preferences, new goals, or revised habits, the corresponding attributes are updated at the end of the session and recorded with explicit operation logs. Consequently, the synthesized trajectory not only maintains long-horizon coherence under anchored knowledge, but also captures how a person's experiences incrementally reshape future events and progressively disclose and update the persona over time.

### 3.3 Question-Answer Pair Synthesis

The entire trajectory construction process can be conceptualized as the generation of a hierarchical tree structure. To fully leverage this hierarchy, KEME employs a bottom-up algorithm for question-answer pair synthesis. Starting from an initial question type tool book $\mathcal{B}_{\text{start}}$, the agent traverses from leaves to the root. At each leaf node, the agent generates question-answer pairs based on the corresponding trajectory segment. At each internal node, the agent synthesizes complex, cross-session questions using child pairs $\mathcal{Q}_{\text{child}}$ from its children as building blocks. Unused child pairs are randomly sampled and passed upward along with the new complex pairs. During synthesis, the agent can dynamically extend the question tool book with new question types.

### 3.4 Quality Control

We implement a multi-staged quality assurance pipeline to ensure the reliability of the synthesized data. We first conduct manual spot-checks on the outputs across different agents. Benefiting from the meticulously fine-tuned prompts and built-in automatic verification of KEME, the quality of generation is consistently high. To further refine the trajectory data, we merge overlapping sessions caused by simultaneous independent events and perform author-led manual inspection. For question-answer pairs, LLMs are leveraged to verify the sufficiency of supporting evidence and identify questions with high similarity to previously assessed pairs. The instances identified as low quality or redundant undergo an LLM-driven rewriting process. Finally, a manual inspection is performed to eliminate any potential artifacts. Figure 4 demonstrates some question-answer pairs in MobileMem.

## 4 Experiments

### 4.1 Experimental Setup

We conduct our experimental evaluation on the MobileMem benchmark to assess a range of memory management approaches. The methods compared include naive RAG Lewis et al. (2020) alongside several specialized memory frameworks: Mem0 Chhikara et al. (2025), LangMem (LangChain, 2025), and A-Mem Xu et al. (2025). To ensure robustness across different model architectures, each approach is implemented using two distinct LLM backbones: GPT-4o-mini OpenAI (2024) and Qwen3-235B-A22B-Instruct-2507 Yang et al. (2025a). All methods utilize all-MiniLM-L6-v2[7] model for text embedding. The number of retrieved memory units is fixed at 30.

[7] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

| Method | Single-Hop | Multi-Hop | Temporal Reasoning | Relationship | Query-Focused Summarization | Adversarial | Others | Overall |
|---|---|---|---|---|---|---|---|---|
| **GPT-4o mini** | | | | | | | | |
| NaiveRAG | 54.15 | 35.29 | 39.13 | 44.74 | 16.44 | 74.19 | 52.89 | 48.95 |
| LangMem | 36.31 | 29.02 | 25.00 | 44.74 | 13.70 | 78.23 | 49.04 | 38.02 |
| A-MEM | 53.08 | 38.04 | 35.87 | 36.84 | 17.81 | 68.55 | 53.85 | 48.13 |
| Mem0 | 38.46 | 27.06 | 17.39 | 34.21 | 10.96 | 78.23 | 46.15 | 37.50 |
| **Qwen3-235B-A22B-Instruct-2507** | | | | | | | | |
| NaiveRAG | 53.54 | 35.29 | 40.22 | 36.84 | 16.44 | 76.61 | 56.73 | 49.03 |
| LangMem | 36.62 | 22.75 | 27.17 | 26.32 | 10.96 | 66.94 | 49.04 | 35.40 |
| A-MEM | 52.31 | 35.69 | 34.78 | 34.21 | 19.18 | 68.55 | 54.81 | 47.31 |
| Mem0 | 43.85 | 31.37 | 25.00 | 39.47 | 15.07 | 78.23 | 37.50 | 41.17 |

Table 1: **End-to-end question-answering performance of baselines.** Two LLM backbones, GPT-4o mini and Qwen3-235B-A22B-Instruct-2507, are used. Due to space limitations, the six most frequent question types are displayed, while the remaining question types are grouped into "Others".

### 4.2 Main Results

As shown in Table 1, a surprising observation is that sophisticated state-of-the-art memory systems struggle to outperform a simple RAG baseline on MobileMem. Even the best-performing approach, RAG, only achieves an average accuracy of approximately 49.03%, failing to reach the 50% threshold. This significantly lower performance compared to existing benchmarks, such as LongMemEval, highlights the substantial challenges posed by the heterogeneous and multi-source nature of our dataset.

Regarding the impact of the LLM backbone, we observe that the underlying model's capability affects the performance of different systems to varying degrees. Notably, the performance variance between Qwen3-235B-A22B-Instruct-2507 and GPT-4o mini is minimal for RAG (a marginal difference of 0.08% on average), whereas it is most pronounced for Mem0. This is because the memory construction process of RAG primarily depends on the fixed retriever rather than the LLM. In contrast, advanced memory frameworks like Mem0 and A-MEM rely heavily on the LLM for complex tasks such as memory extraction and condensation making the LLM's reasoning capability a critical bottleneck for these systems.

Looking at performance across question categories, all baselines consistently achieve their highest performance on adversarial questions. This suggests that existing memory systems, benefiting from the reduced hallucination tendencies of modern LLMs, possess a robust capability to identify unanswerable queries. RAG demonstrates its strength in single-hop and relationship-related questions, where direct retrieval often suffices for a correct response. However, A-MEM outperforms RAG in multi-hop reasoning and query-focused summarization. This advantage stems from A-MEM's graph-structured memory architecture, which dynamically establishes links between disparate memory units and allows the memory content to evolve. Such a mechanism allows the system to integrate information from multiple sources in a more effective and principled manner than non-graph-structured baselines.

### 4.3 Analysis

Our error analysis focuses on the baselines using GPT-4o mini as the backbone. **We categorize memory failures into three distinct stages: memory construction, memory retrieval, and response generation.** We observe that for complex memory systems like Mem0 and LangMem, memory construction is the primary bottleneck, accounting for 71.02% and 68.96% of total failures, respectively. In contrast, A-MEM exhibits a significantly lower construction error rate of 34.63%. This disparity primarily stems from the fact that Mem0 and LangMem support memory deletion operations. The erroneous removal of information during the update process leads to the permanent loss of salient details. A-MEM avoids this pitfall as it does not implement a deletion mechanism. In sharp contrast, vanilla RAG exhibits a 0% memory construction error, as it directly indexes raw interaction logs. Consequently, for both A-MEM and RAG, the burden shifts to the retrieval stage, which becomes the dominant failure mode, accounting for 59.31% and 94.13% of their errors, respectively. Ultimately, compared to vanilla RAG, other memory systems introduce an extraction module for information processing, which inherently increases the risk of memory construction errors.

Finally, across almost all baselines, the response stage consistently accounts for the smallest error fraction ($< 6.1\%$). This suggests that once the relevant context is correctly retrieved, the underlying LLMs are generally robust enough to generate accurate answers, further confirming that the core challenge of long-term AI memory lies in the efficient management and retrieval of information rather than the final reasoning step.

## 5 Related Work

**LLM Memory.** Existing work on LLM memory approaches Liu et al. (2025a); Mei et al. (2025); Zhang et al. (2024); Hu et al. (2025b) can be broadly divided into non-parametric and parametric approaches. Non-parametric memory equips LLMs with external storage, allowing them to read from and write to this storage for managing and utilizing memory LangChain (2025); Zhong et al. (2024); Packer et al. (2023); Chhikara et al. (2025); Xu et al. (2025); Li et al. (2025); Kang et al. (2025); Wang & Chen (2025); Fang et al. (2025b); Ouyang et al. (2025); Fang et al. (2025a); Wang et al. (2025a); Latimer et al. (2025); Yan et al. (2025a); Wu et al. (2025c); Du et al. (2025a). Parametric memory instead performs read–write operations on the model's own internal states Wang et al. (2024); Behrouz et al. (2024); Sun et al. (2025b); Wang et al. (2025b); He et al. (2025); Lin et al. (2025); Bini et al. (2025); Wei et al. (2025). Recently, several studies have applied reinforcement learning to improve memory management and usage in both non-parametric Yan et al. (2025c); Wang et al. (2025c) and parametric settings Zhou et al. (2025); Zhang et al. (2025b).

**Memory Benchmark.** LoCoMo Maharana et al. (2024) serves as one of the earliest and most widely used benchmarks for evaluating long-term memory in LLMs. Building on this foundation, later benchmarks such as LongMemEval Wu et al. (2025a) and MemBench Tan et al. (2025) broaden the evaluation scope by adding new types of question–answer pairs and lengthening user trajectories. Another line of work improves the quality of trajectories, aiming for higher realism Kim et al.; Lee et al. (2025) or greater diversity Hu et al. (2025a). Notably, ScreenshotVQA Wang & Chen (2025) models each user trajectory as a stream of screenshots. Complementary datasets such as PersonaMem Jiang et al. (2025a) and MemGuide Du et al. (2025b) shift the focus to specific application scenarios. Furthermore, EgoLifeQA Yang et al. (2025b) and TeleEgo Yan et al. (2025b) emphasize the evaluation of memory within multimodal scenarios. More recently, HaluMem Chen et al. (2025) advances the evaluation methodology by introducing operation-level hallucination analysis, enabling a more fine-grained assessment of memory performance. To the best of our knowledge, no existing benchmark systematically evaluates memory in multi-source contexts that combine participation-based user–assistant interactions with observation-based information arising from user engagements with third-party applications.

## 6 Limitations

This work has several limitations. First, while MobileMem is designed to model long-horizon and heterogeneous memory usage, the current benchmark remains limited in temporal scale and overall size, and does not yet incorporate rich multimodal signals such as images, audio, or sensor data. Extending the duration, scale, and modality coverage of MobileMem is an important direction for future work, and we plan to continuously expand and maintain the benchmark to better reflect real-world assistant usage. Second, our empirical evaluation covers a limited set of LLM backbones and memory systems. Although the selected models are representative of current approaches, broader evaluation across a wider range of architectures and memory designs would provide a more comprehensive understanding of memory system behavior. We leave this to future work.

## 7 Conclusion

We introduce MobileMem, a benchmark for evaluating memory frameworks in real-world mobile environments. Unlike prior benchmarks, MobileMem focuses on long-horizon, real-world mobile scenarios. Our results show that existing memory systems struggle to perform well, underscoring the need for powerful long-term memory frameworks.

## ETHICS STATEMENT

The development of MobileMem adheres to strict ethical standards regarding data collection and dissemination. All data used in MobileMem are obtained under signed agreements, with explicit user consent for public release. Sensitive information is rigorously anonymized or modified to prevent identification of individuals.

In addition to real-world data, MobileMem includes synthetic behavior traces generated with LLMs to augment coverage of plausible user actions. All such synthetic data are carefully reviewed and curated by humans to ensure quality, realism, and consistency with the grounded user personas.

To ensure privacy and compliance with ethical standards, all user-persona grounding interviews and application interaction traces are reviewed by the Institutional Review Board (IRB) of the data-providing organization. MobileMem is released under the MIT License, with clear documentation and anonymization measures, and users are encouraged to uphold these privacy safeguards in downstream research.

By combining careful anonymization, human-verified synthetic data, IRB oversight, and controlled release, we aim to facilitate research on long-term, multi-source memory in LLMs while maintaining strong ethical standards and respect for user privacy.

## REFERENCES

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

Massimo Bini, Ondrej Bohdal, Umberto Michieli, Zeynep Akata, Mete Ozay, and Taha Yusuf Ceritli. Memlora: Distilling expert adapters for on-device memory systems. 2025. URL https://api.semanticscholar.org/CorpusID:283557414.

Hongru Cai, Yongqi Li, Wenjie Wang, Fengbin Zhu, Xiaoyu Shen, Wenjie Li, and Tat-Seng Chua. Large language models empowered personalized web agents. In *Proceedings of the ACM on Web Conference 2025*, pp. 198–215, 2025.

Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *ArXiv*, abs/2406.20094, 2024. URL https://api.semanticscholar.org/CorpusID:270845490.

Keshigeyan Chandrasegaran, Agrim Gupta, Lea M. Hadzic, Taran Kota, Jimming He, Cristóbal Eyzaguirre, Zane Durante, Manling Li, Jiajun Wu, and Li Fei-Fei. Hourvideo: 1-hour video-language understanding. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/5f2809607f692d79a01c05c43d702883-Abstract-Datasets_and_Benchmarks_Track.html.

Ding Chen, Simin Niu, Kehang Li, Peng Liu, Xiangping Zheng, Bo Tang, Xinchi Li, Feiyu Xiong, and Zhiyu Li. Halumem: Evaluating hallucinations in memory systems of agents. *ArXiv*, abs/2511.03506, 2025. URL https://api.semanticscholar.org/CorpusID:282758378.

Jin Chen, Zheng Liu, Xu Huang, Chenwang Wu, Qi Liu, Gangwei Jiang, Yuanhao Pu, Yuxuan Lei, Xiaolong Chen, Xingmei Wang, et al. When large language models meet personalization: Perspectives of challenges and opportunities. *World Wide Web*, 27(4):42, 2024.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *ArXiv*, abs/2504.19413, 2025. URL https://api.semanticscholar.org/CorpusID:278165315.

Xingbo Du, Loka Li, Duzhen Zhang, and Le Song. Memr$^3$: Memory retrieval via reflective reasoning for llm agents. *arXiv preprint arXiv:2512.20237*, 2025a.

Yiming Du, Bingbing Wang, Yang He, Bin Liang, Baojun Wang, Zhongyang Li, Lin Gui, Jeff Z. Pan, Ruifeng Xu, and Kam-Fai Wong. Memguide: Intent-driven memory selection for goal-oriented multi-session llm agents. 2025b. URL https://api.semanticscholar.org/CorpusID:278911777.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. Lightmem: Lightweight and efficient memory-augmented generation. *CoRR*, abs/2510.18866, 2025a. doi: 10.48550/ARXIV.2510.18866. URL https://doi.org/10.48550/arXiv.2510.18866.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. *ArXiv*, abs/2508.06433, 2025b. URL https://api.semanticscholar.org/CorpusID:280561810.

Dawei Gao, Zitao Li, Yuexiang Xie, Weirui Kuang, Liuyi Yao, Bingchen Qian, Zhijian Ma, Yue Cui, Haohao Luo, Shen Li, Lu Yi, Yi Yu, Shiqi He, Zhiling Luo, Wenmeng Zhou, Zhicheng Zhang, Xuguang He, Ziqian Chen, Weikai Liao, Farruh Isakulovich Kushnazarov, Yaliang Li, Bolin Ding, and Jingren Zhou. Agentscope 1.0: A developer-centric framework for building agentic applications. *ArXiv*, abs/2508.16279, 2025. URL https://api.semanticscholar.org/CorpusID:280708369.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Yuanzhuo Wang, and Jian Guo. A survey on llm-as-a-judge. *ArXiv*, abs/2411.15594, 2024. URL https://api.semanticscholar.org/CorpusID:274234014.

Zifan He, Yingqi Cao, Zongyue Qin, Neha Prakriya, Yizhou Sun, and Jason Cong. HMT: hierarchical memory transformer for efficient long context language processing. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pp. 8068–8089. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.NAACL-LONG.410. URL https://doi.org/10.18653/v1/2025.naacl-long.410.

Yuanzhe Hu, Yu Wang, and Julian J. McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. *CoRR*, abs/2507.05257, 2025a. doi: 10.48550/ARXIV.2507.05257. URL https://doi.org/10.48550/arXiv.2507.05257.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, Senjie Jin, Jiejun Tan, Yanbin Yin, Jiongnan Liu, Zeyu Zhang, Zhongxiang Sun, Yutao Zhu, Hao Sun, Boci Peng, Zhenrong Cheng, Xuanbo Fan, Jiaxin Guo, Xinlei Yu, Zhenhong Zhou, Zewen Hu, Jiahao Huo, Junhao Wang, Yuwei Niu, Yu Wang, Zhenfei Yin, Xiaobin Hu, Yue Liao, Qiankun Li, Kun Wang, Wangchunshu Zhou, Yixin Liu, Dawei Cheng, Qi Zhang, Tao Gui, Shirui Pan, Yan Zhang, Philip Torr, Zhicheng Dou, Ji-Rong Wen, Xuanjing Huang, Yu-Gang Jiang, and Shuicheng Yan. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*, 2025b.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo Jose Taylor, and Dan Roth. Know me, respond to me: Benchmarking llms for dynamic user profiling and personalized responses at scale. *ArXiv*, abs/2504.14225, 2025a. URL https://api.semanticscholar.org/CorpusID:277955197.

Hang Jiang, Xiajie Zhang, Xubo Cao, Cynthia Breazeal, Deb Roy, and Jad Kabbara. Personallm: Investigating the ability of large language models to express personality traits. In *Findings of the association for computational linguistics: NAACL 2024*, pp. 3605–3627, 2024.

Hongda Jiang, Xinyuan Zhang, Siddhant Garg, Rishab Arora, Shiun-Zu Kuo, Jiayang Xu, Ankur Bansal, Christopher Brossman, Yue Liu, Aaron Colak, Ahmed Aly, Anuj Kumar, and Xin Dong. Memory-qa: Answering recall questions based on multimodal memories. *ArXiv*, abs/2509.18436, 2025b. URL https://api.semanticscholar.org/CorpusID:281495947.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory os of ai agent. *ArXiv*, abs/2506.06326, 2025. URL https://api.semanticscholar.org/CorpusID:279250574.

Jiho Kim, Woosog Chay, Hyeonji Hwang, Daeun Kyung, Hyunseung Chung, Eunbyeol Cho, Yohan Jo, Edward Choi, and Snu. Dialsim: A real-time simulator for evaluating long-term multi-party dialogue understanding of conversation systems. URL https://api.semanticscholar.org/CorpusID:276409000.

LangChain. Langmem sdk for agent long-term memory, 2025. URL https://blog.langchain.com/langmem-sdk-launch/.

Chris Latimer, Nicoló Boschi, Andrew Neeser, Chris Bartholomew, Gaurav Srivastava, Xuan Wang, and Naren Ramakrishnan. Hindsight is 20/20: Building agent memory that retains, recalls, and reflects, 2025. URL https://arxiv.org/abs/2512.12818.

Dong-Ho Lee, Adyasha Maharana, Jay Pujara, Xiang Ren, and Francesco Barbieri. Realtalk: A 21-day real-world dataset for long-term conversation. *ArXiv*, abs/2502.13270, 2025. URL https://api.semanticscholar.org/CorpusID:276450264.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401, 2020. URL https://api.semanticscholar.org/CorpusID:218869575.

Yanda Li, Chi Zhang, Wenjia Jiang, Wanqi Yang, Bin Fu, Pei Cheng, Xin Chen, Ling Chen, and Yunchao Wei. Appagent v2: Advanced agent for flexible mobile interactions. *arXiv preprint arXiv:2408.11824*, 2024.

Zhiyu Li, Shichao Song, Hanyu Wang, Simin Niu, Ding Chen, Jiawei Yang, Chenyang Xi, Huayi Lai, Jihao Zhao, Yezhaohui Wang, Junpeng Ren, Zehao Lin, Jiahao Huo, Tianyi Chen, Kai Chen, Ke-Rong Li, Zhiqiang Yin, Qingchen Yu, Bo Tang, Hongkang Yang, Zhiyang Xu, and Feiyu Xiong. Memos: An operating system for memory-augmented generation (mag) in large language models. *ArXiv*, abs/2505.22101, 2025. URL https://api.semanticscholar.org/CorpusID:278960153.

Jessy Lin, Luke S. Zettlemoyer, Gargi Ghosh, Wen tau Yih, Aram H. Markosyan, Vincent-Pierre Berges, and Barlas Ouguz. Continual learning via sparse memory finetuning. *ArXiv*, abs/2510.15103, 2025. URL https://api.semanticscholar.org/CorpusID:282203348.

Bang Liu, Xinfeng Li, Jiayi Zhang, Jinlin Wang, Tanjin He, Sirui Hong, Hongzhang Liu, Shaokun Zhang, Kaitao Song, Kunlun Zhu, et al. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*, 2025a.

Jiahong Liu, Zexuan Qiu, Zhongyang Li, Quanyu Dai, Wenhao Yu, Jieming Zhu, Minda Hu, Menglin Yang, Tat-Seng Chua, and Irwin King. A survey of personalized large language models: Progress and future directions. *arXiv preprint arXiv:2502.11528*, 2025b.

Lin Long, Yichen He, Wen song Ye, Yiyuan Pan, Yuan Lin, Hang Li, Junbo Zhao, and Wei Li. Seeing, listening, remembering, and reasoning: A multimodal agent with long-term memory. *ArXiv*, abs/2508.09736, 2025. URL https://api.semanticscholar.org/CorpusID:280642200.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 13851–13870. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.747. URL https://doi.org/10.18653/v1/2024.acl-long.747.

Lingrui Mei, Jiayu Yao, Yuyao Ge, Yiwei Wang, Baolong Bi, Yujun Cai, Jiazhi Liu, Mingyu Li, Zhong-Zhi Li, Duzhen Zhang, et al. A survey of context engineering for large language models. *arXiv preprint arXiv:2507.13334*, 2025.

OpenAI. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

OpenAI. Introducing gpt-4.1 in the api, 2025a. URL https://openai.com/index/gpt-4-1/.

OpenAI. Introducing gpt-5.2, 2025b. URL https://openai.com/index/gpt-5-2/.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. Reasoningbank: Scaling agent self-evolving with reasoning memory. *ArXiv*, abs/2509.25140, 2025. URL https://api.semanticscholar.org/CorpusID:281674540.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph Gonzalez. Memgpt: Towards llms as operating systems. *ArXiv*, abs/2310.08560, 2023. URL https://api.semanticscholar.org/CorpusID:263909014.

Iryna Pentina, Lixuan Zhang, Hatem Bata, and Ying Chen. Exploring privacy paradox in information-sensitive mobile app adoption: A cross-cultural comparison. *Computers in Human Behavior*, 65:409–419, 2016.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Jessica Su, Anshika Shukla, Sharad Goel, and Arvind Narayanan. De-anonymizing web browsing data with social networks. *Proceedings of the 26th International Conference on World Wide Web*, 2017. URL https://api.semanticscholar.org/CorpusID:2921869.

Chenkai Sun, Ke Yang, Revanth Gangi Reddy, Yi Fung, Hou Pong Chan, Kevin Small, ChengXiang Zhai, and Heng Ji. Persona-db: Efficient large language model personalization for response prediction with collaborative data refinement. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 281–296, 2025a.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): Rnns with expressive hidden states. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net, 2025b. URL https://openreview.net/forum?id=wXfuOj9C7L.

Haoran Tan, Zeyu Zhang, Chen Ma, Xu Chen, Quanyu Dai, and Zhenhua Dong. Membench: Towards more comprehensive evaluation on the memory of llm-based agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 19336–19352. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-acl.989/.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. A survey on (m)llm-based gui agents. *ArXiv*, abs/2504.13865, 2025. URL https://api.semanticscholar.org/CorpusID:277955123.

Zhen Tu, Runtong Li, Yong Li, Gang Wang, Di Wu, Pan Hui, Li Su, and Depeng Jin. Your apps give you away: Distinguishing mobile users by their app usage fingerprints. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(3):1–23, 2018.

Volcengine. Minecontext: Create with context, clarity from chaos, 2025. URL https://github.com/volcengine/MineContext.

Piaohong Wang, Motong Tian, Jiaxian Li, Yuan Liang, Yuqing Wang, Qianben Chen, Tiannan Wang, Zhicong Lu, Jiawei Ma, Yuchen Eleanor Jiang, and Wangchunshu Zhou. O-mem: Omni memory system for personalized, long horizon, self-evolving agents. 2025a. URL https://api.semanticscholar.org/CorpusID:283073241.

Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet A. Orgun, and Defu Lian. A survey on session-based recommender systems. *ACM Comput. Surv.*, 54(7):154:1–154:38, 2022. doi: 10.1145/3465401. URL https://doi.org/10.1145/3465401.

Yu Wang and Xi Chen. Mirix: Multi-agent memory system for llm-based agents. *arXiv preprint arXiv:2507.07957*, 2025.

Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, Jingbo Shang, and Julian J. McAuley. MEMORYLLM: towards self-updatable large language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=p0lKWzdikQ.

Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian J. McAuley, Dan Gutfreund, Rogério Feris, and Zexue He. M+: extending memoryllm with scalable long-term memory. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net, 2025b. URL https://openreview.net/forum?id=OcqbkROe8J.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-$\alpha$: Learning memory construction via reinforcement learning. *ArXiv*, abs/2509.25911, 2025c. URL https://api.semanticscholar.org/CorpusID:281682069.

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, Yu Su, Upol Ehsan, Malihe Alikhani, Toby Jia-Jun Li, Lydia Chilton, and Dakuo Wang. Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation. *ArXiv*, abs/2506.05606, 2025d. URL https://api.semanticscholar.org/CorpusID:279244562.

Rubin Wei, Jiaqi Cao, Jiarui Wang, Jushi Kai, Qipeng Guo, Bowen Zhou, and Zhouhan Lin. Mlp memory: A retriever-pretrained memory for large language models. 2025. URL https://api.semanticscholar.org/CorpusID:281658735.

Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. Autodroid: Llm-powered task automation in android. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, pp. 543–557, 2024.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025a. URL https://openreview.net/forum?id=pZiyCaVuti.

Wenyi Wu, Kun Zhou, Ruoxin Yuan, Vivian Yu, Stephen Wang, Zhiting Hu, and Biwei Huang. Auto-scaling continuous memory for gui agent. *ArXiv*, abs/2510.09038, 2025b. URL https://api.semanticscholar.org/CorpusID:282055682.

Yaxiong Wu, Yongyue Zhang, Sheng Liang, and Yong Liu. Sgmem: Sentence graph memory for long-term conversational agents. *arXiv preprint arXiv:2509.21212*, 2025c.

Jing Xu, Arthur Szlam, and Jason Weston. Beyond goldfish memory: Long-term open-domain conversation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 5180–5197. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.356. URL https://doi.org/10.18653/v1/2022.acl-long.356.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *ArXiv*, abs/2502.12110, 2025. URL https://api.semanticscholar.org/CorpusID:276421617.

BY Yan, Chaofan Li, Hongjin Qian, Shuqi Lu, and Zheng Liu. General agentic memory via deep research. *arXiv preprint arXiv:2511.18423*, 2025a.

Jiaqi Yan, Ruilong Ren, Jingren Liu, Shuning Xu, Ling Wang, Yiheng Wang, Yun Wang, Long Zhang, Xiangyu Chen, Changzhi Sun, Jixiang Luo, Dell Zhang, Hao Sun, Chi Zhang, and Xuelong Li. Teleego: Benchmarking egocentric ai assistants in the wild. *ArXiv*, abs/2510.23981, 2025b. URL https://api.semanticscholar.org/CorpusID:282401233.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Hinrich Schutze, Volker Tresp, and Yunpu Ma. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *ArXiv*, abs/2508.19828, 2025c. URL https://api.semanticscholar.org/CorpusID:280918480.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jian Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025a. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Jingkang Yang, Shuai Liu, Hongming Guo, Yuhao Dong, Xiamengwei Zhang, Sicheng Zhang, Pengyun Wang, Zitang Zhou, Binzhu Xie, Ziyue Wang, Bei Ouyang, Zhengyu Lin, Marco Cominelli, Zhongang Cai, Bo Li, Yuanhan Zhang, Peiyuan Zhang, Fangzhou Hong, Joerg Widmer, Francesco Gringoli, Lei Yang, and Ziwei Liu. Egolife: Towards egocentric life assistant. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 28885–28900. Computer Vision Foundation / IEEE, 2025b. doi: 10.1109/CVPR52734.2025.02690. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yang_EgoLife_Towards_Egocentric_Life_Assistant_CVPR_2025_paper.html.

Jiabo Ye, Xi Zhang, Haiyang Xu, Haowei Liu, Junyang Wang, Zhaoqing Zhu, Ziwei Zheng, Feiyu Gao, Junjie Cao, Zhengxi Lu, Jitong Liao, Qi Zheng, Fei Huang, Jingren Zhou, and Ming Yan. Mobile-agent-v3: Fundamental agents for gui automation. *ArXiv*, abs/2508.15144, 2025. URL https://api.semanticscholar.org/CorpusID:280699844.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yanda Li, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users. In Naomi Yamashita, Vanessa Evers, Koji Yatani, Sharon Xianghua Ding, Bongshin Lee, Marshini Chetty, and Phoebe O. Toups Dugas (eds.), *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems, CHI 2025, YokohamaJapan, 26 April 2025- 1 May 2025*, pp. 70:1–70:20. ACM, 2025a. doi: 10.1145/3706598.3713600. URL https://doi.org/10.1145/3706598.3713600.

Gui-Min Zhang, Muxin Fu, and Shuicheng Yan. Memgen: Weaving generative latent memory for self-evolving agents. *ArXiv*, abs/2509.24704, 2025b. URL https://api.semanticscholar.org/CorpusID:281676243.

Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R. Tomz, Christopher D. Manning, and Weiyan Shi. Verbalized sampling: How to mitigate mode collapse and unlock llm diversity. *ArXiv*, abs/2510.01171, 2025c. URL https://api.semanticscholar.org/CorpusID:281705900.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model based agents, 2024. URL https://arxiv.org/abs/2404.13501.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 19724–19731. AAAI Press, 2024. doi: 10.1609/AAAI.V38I17.29946. URL https://doi.org/10.1609/aaai.v38i17.29946.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *ArXiv*, abs/2506.15841, 2025. URL https://api.semanticscholar.org/CorpusID:279465470.

# A Dataset

## A.1 Mobile Applications in MobileMem

MobileMem involves 7 representative mobile apps. Most of them are system-level or pre-installed apps that naturally support persistent, long-term user memory across daily activities. Table 2 outlines the functionality of each app.

| App | Description |
|---|---|
| **Bill** | A financial management tool that automatically records income and expenditure transactions, providing automated categorization for different payments. |
| **Voice Recorder** | An audio logging tool for capturing events and spontaneous thoughts via voice, facilitating on-the-fly information recording. |
| **Screen Memo** | A visual memory utility that automatically extracts and archives on-screen text content upon a user-triggered activation. |
| **Document** | A comprehensive document utility capable of scanning, viewing, editing, translating, converting formats, and storing files. |
| **Note** | A digital notebook for documenting personal thoughts, reflections, daily observations, and cherished moments. |
| **Calendar** | A time-management application that enables users to schedule events, organize appointments, and plan future activities. |
| **To-Do List** | A task management tool for organizing pending duties and setting timely reminders to track deadlines and completion status. |

Table 2: The descriptions and functionalities of mobile applications included in MobileMem.

## A.2 App-Specific Perception Protocols and Message Templates

To construct our dataset, we define a specific perception protocol for each application to ensure that the memory system receives structured, high-quality information. Each application implements a lightweight perception module that processes interactions locally and generates a natural-language message based on a predefined template. Below, we detail the perception protocols and the corresponding message templates for the applications used in MobileMem.

### A.2.1 Voice Recorder

**Protocol:** The Voice Recorder app integrates a lightweight on-device Automatic Speech Recognition (ASR) system. Upon the completion of a recording, the app calculates the audio duration and transcribes the speech content into text.

**Message Template:**

```
The user records a voice memo (Duration: [Duration]
seconds): [Transcript]
```

#### A.2.2 CALENDAR

**Protocol:** The Calendar app triggers a message transmission whenever a user creates an event, extracting temporal details and reminder settings.

**Message Template:**

```
The user records an event ``[Title]'' in the calendar.
The details are as follows:
Event Description: [Description]
Start Time: [Start Time]
End Time: [End Time]
Reminder: [Reminder Details]
```

#### A.2.3 TO-DO LIST

**Protocol:** The To-Do List app employs a state-aware protocol using rule-based triggers. It detects whether a task is being created or marked as completed and generates the message accordingly.

**Message Template:**

```
The user [adds / completes] a to-do item: ``[Title]''
Details: [Details]
Due: [Due Date]
Priority: [Priority Level]
Status: [Finished / Unfinished]
```

#### A.2.4 BILL

**Protocol:** The Bill app extracts transaction details and utilizes a classification model to determine the specific category of the expenditure (e.g., medical, dining). It automatically maps these classified categories to natural language descriptions and formats monetary values with their corresponding currencies.

**Message Template:**

```
A transaction record is generated between user and
[Merchant].
Product: [Product Name]
Amount: [Value] [Currency]
Transaction Type: [Income / Expense]
Category: [Category Name]
Payment Method: [Payment Method]
```

#### A.2.5 NOTE

**Protocol:** The Note app perception module triggers when a user saves a text note, directly transmitting the full content and title to the memory system without summarization.

**Message Template:**

```
The user creates a note: ``[Title]''
Content: [Content]
```

#### A.2.6 DOCUMENT

**Protocol:** The Document perception module triggers when a user saves a document file. It automatically extracts file metadata (such as page count and format) and parses the document content.

**Message Template:**

```
The user saves a document titled ``[Title]''.
Content:
[Content] Topic:  [Topic]
Format:  [File Format]
Pages:  [Page Count]
```

### A.2.7 SCREEN MEMO

**Protocol:** The Screen Memo perception module is manually triggered when the user presses a specific button. It utilizes a system composed of multiple neural networks to perform Optical Character Recognition (OCR) on the current screen. The system extracts the textual content while preserving its original spatial layout. Note that in our dataset, all visual imagery is filtered out.

**Message Template:**

```
The user views the following content on phone:
[Content Text]
```

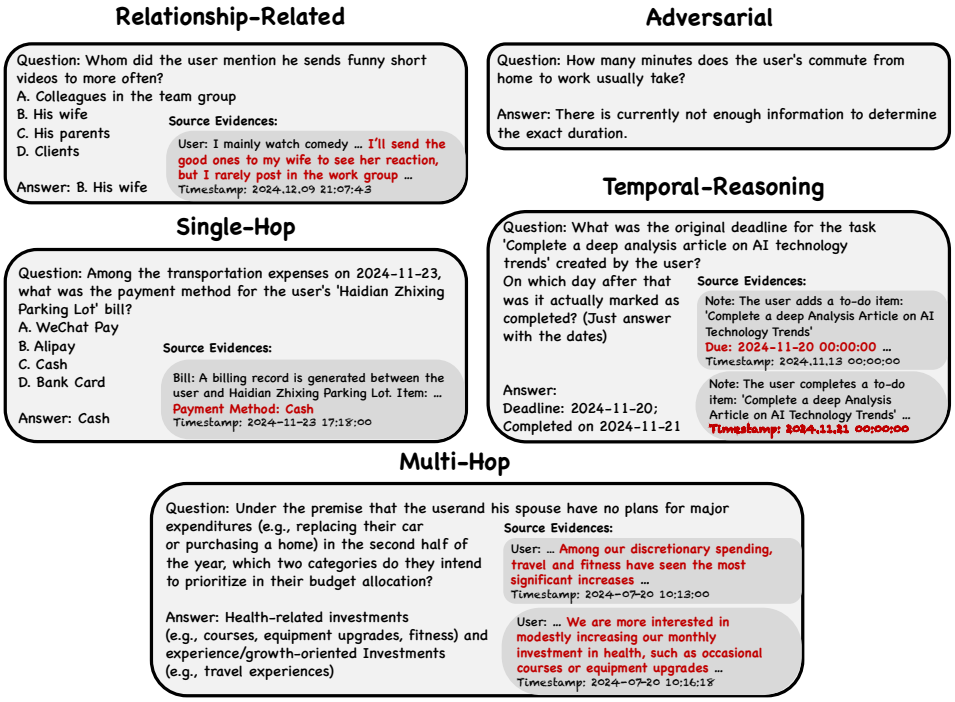


Figure 4: **Question-answer pair examples in MobileMem.** We design six main types of question-answer pairs to evaluate AI memory system performances in *single-hop recall*, *multi-hop reasoning*, *temporal reasoning*, *relationship understanding*, *query-focused summarization*, and *unanswerable question recognization*. The question formats include *single-choice*, *multiple-choice*, and *open-ended*. Due to space limitations, query-focused summarization question-answer pairs are not displayed in the figure.

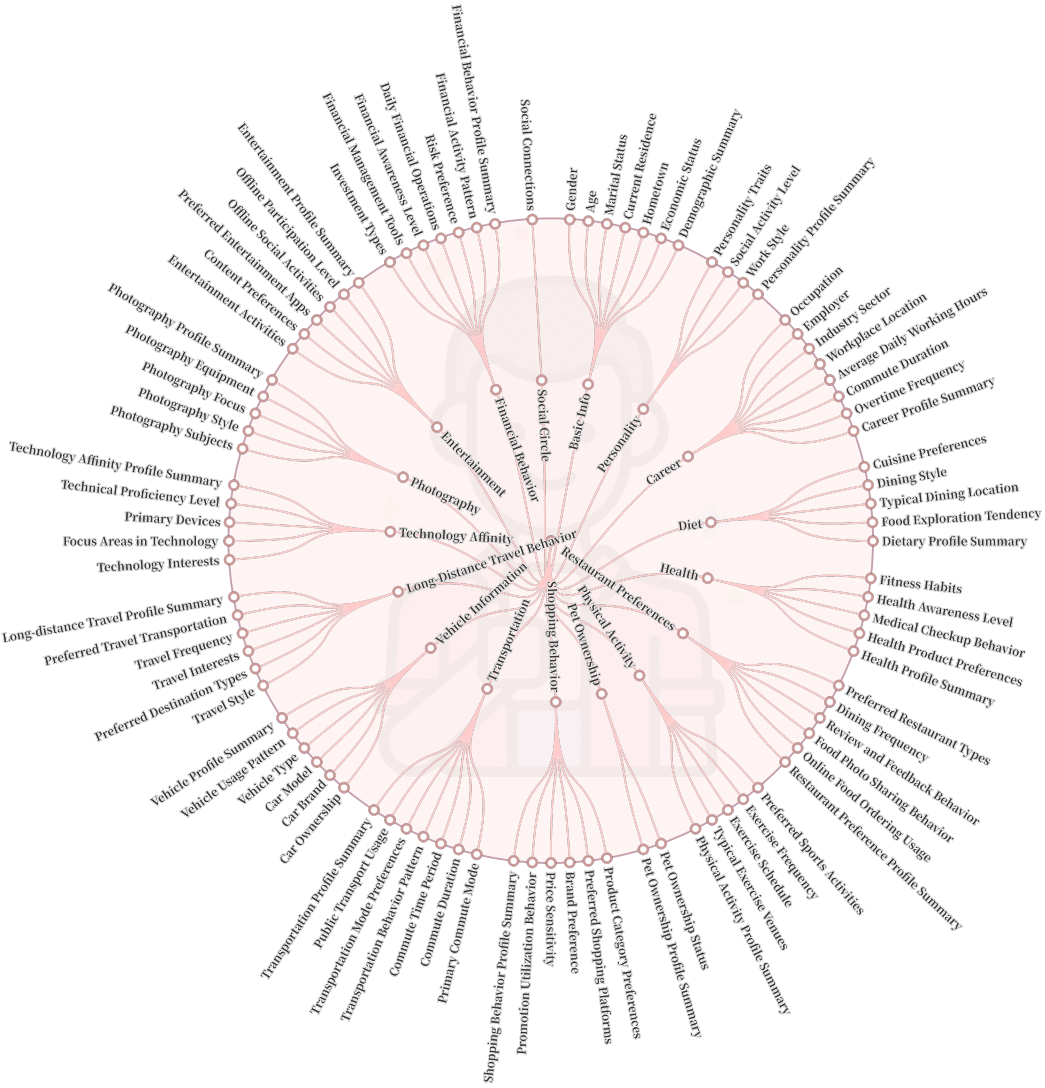


Figure 5: **The persona profile schema used in MobileMem.** This schema provides a holistic and multi-dimensional representation of a user by integrating 17 foundational aspects. Each aspect encompasses a diverse set of specific attributes, providing a granular characterization that ensures the synthesized trajectories are grounded in realistic and consistent behavioral logic.

# B Implementation Details of KEME

## B.1 Data Models and Tools

To ensure its reliability, KEME represents all intermediate artifacts as Pydantic models, enabling explicit validation and ensuring strictly structured outputs[8] across the entire pipeline. Below, we detail the data models and related tools used in KEME.

### B.1.1 Person

**Function:** `Person` is the global state of a trajectory. It specifies the global time horizon and maintains an evolving global user profile composed of multiple dimension models. The model also stores the knowledge anchors associated with this user.

**Related Tools:** The person profile is updated through session-level tools that (i) revise a dimension attribute while recording operation logs and timestamps, and (ii) link user or system messages to specific attributes as evidence, thereby making attribute disclosure traceable.

### B.1.2 Requirement

**Function:** `Requirement` represents an explicit, provenance-tracked constraint that conditions downstream synthesis decisions. Beyond simply propagating context, it serves as a minimal, actionable specification of (i) what must hold in future expansions (e.g., logical constraints), (ii) what goals or follow-ups should be introduced, and (iii) what user-state changes are expected to occur after certain experiences. Each requirement includes an identifier and name, a detailed description, a source field that records its origin (person profile, an upstream event[9], or an agent[10]), as well as a short source evidence snippet and optional side notes. This design makes constraints inheritable, and auditable, enabling top-down guidance and refinement-driven evolution without requiring full-context replay at every step.

**Related Tools:** Requirements are not edited through a standalone tool. Instead, they are created, copied, and revised when constructing or refining a temporal event graph.

### B.1.3 Event

**Function:** `Event` is a time-bounded semantic unit in the trajectory hierarchy. It constrains downstream expansion semantically through its title, summary, and requirements (`Requirement`), and temporally through its time span. An event may also carry knowledge anchors and a compatibility context. Both act as additional semantic constraints by preserving anchored facts that future expansions must stay compatible with. During synthesis, the state field indicates the expansion status, and the output field stores the expansion result as either a leaf session or a child temporal event graph.

**Related Tools:** Events are created and revised indirectly through temporal event graph tools used in graph construction and refinement. Grounding-related tools assign knowledge anchors to events and update the compatibility context. Event expansion tools write the expansion result to the event output and update the event state.

### B.1.4 Edge

**Function:** `Edge` encodes a directed dependency between two sibling events, with a concise `name` and references to the source and target event (`Event`). Edges impose temporal ordering and define the expansion order.

[8] https://platform.openai.com/docs/guides/structured-outputs?api-mode=chat

[9] An upstream event refers to either an ancestor event in the hierarchy or a predecessor sibling event connected by a dependency edge in the temporal event graph. In the latter case, the edge indicates that outcomes or constraints from the predecessor should constrain the successor.

[10] Agent-sourced requirements are additional constraints intentionally introduced by the synthesis agent to improve coherence and evaluation difficulty, such as narrative requirements, anticipated person-state changes, emergent follow-up goals, logical implications, or conflict resolutions.

**Related Tools:** Edges are manipulated indirectly through tools that construct or refine a temporal event graph. These tools allow for adding, revising, and removing edges as part of maintaining a coherent acyclic dependency structure.

#### B.1.5 TEMPORAL EVENT GRAPH

**Function:** `Temporal Event Graph` organizes a set of sibling events into a directed acyclic graph within the same hierarchy level. It provides the structural constraints that govern synthesis, including the event (`Event`) set, dependency edges (`Edge`), and a graph-level state that controls the workflow from graph initialization to session grounding and event expansion. The acyclic dependency structure defines valid expansion order and supports topological selection of the next event to expand.

**Related Tools:** The temporal event graph is created and maintained through graph construction and refinement tools. These tools initialize events and edges, revise unexpanded events and dependencies. During synthesis, grounding-related tools allocate knowledge anchors to compatible events within the graph, and event expansion tools iteratively expand events until the graph is completed.

#### B.1.6 MESSAGE

**Function:** `Message` is the atomic interaction record. It stores a unique identifier, sender name, role, natural-language content, and a timestamp. It also includes optional metadata which can attach auxiliary information.

**Related Tools:** Messages are not created in isolation. They are instantiated as part of session creation tools. User or system messages are eligible to be linked as evidence for persona attributes through session-level linking tools.

#### B.1.7 SESSION

**Function:** `Session` is the leaf-level semantic unit of the trajectory. A session includes a unique identifier, an optional event identifier linking it to its parent, and an ordered list of messages (`Message`). Sessions also carry optional side notes for explaining narrative intent and outcomes.

**Related Tools:** Session tools create a session under the parent semantic and temporal constraints, update the person profile, and link selected user or system messages to persona attributes as evidence.

### B.2 RULES AND FEEDBACK ENGINEERING

To further improve the robustness, controllability, effectiveness, and traceability of KEME, we build explicit rules and human-readable feedback into every tool invocation. These rules constrain the agent's action space and provide actionable diagnostics when violations occur. For example, when initializing a temporal event graph for a parent node, the system validates requirement provenance. It also checks that every event's time interval is fully contained within the parent's time span. During graph edits, newly added dependency edges are checked for strict temporal consistency enforcing that the source event ends no later than the target event begins. Events that have already been expanded are treated as immutable to prevent inconsistencies with their expansion results. Moreover, during refinement, the system prohibits deleting events that contain grounded knowledge anchors, ensuring anchored evidence is preserved. We further bound graph complexity by enforcing a minimum and maximum event count ($m_{\min}$, $m_{\max}$) in graph-related tools, preventing both underspecified and over-fragmented graph structures.

Whenever a rule is violated, the tool returns clear error messages that pinpoint what goes wrong and why, enabling reliable self-correction and stabilizing the overall synthesis process. Without these rules and feedback, weaker models can easily fall into degenerate loops by repeatedly producing the same outputs, while even stronger models may occasionally generate results that violate the constraints specified in the prompts.

### B.3 Expansion Strategies

In the main text, we describe a depth-based expansion strategy where the transition from event graphs to concrete sessions is governed by a maximum hierarchy depth $d_{\max}$. To further enhance the flexibility and realism of KEME across diverse temporal scales, we provide three alternative strategies to determine when an event node $n$ should be realized as a session $\mathcal{S}^{(n)}$ rather than being further decomposed into a sub-event graph $\mathcal{G}^{(n)}$.

#### B.3.1 Deterministic Duration-Based Strategy

This strategy is grounded in the heuristic that human interactions have an intrinsic temporal granularity. If an event's duration $t$ falls below a predefined minimum threshold $T_{\min}$, it is considered too granular to be further subdivided and is directly converted into a concrete session.

#### B.3.2 Stochastic Probability-Based Strategy

Recognizing that the transition from broad life events to specific interactions is often non-linear and fuzzy, we introduce a stochastic transition mechanism. For events with a duration $t$ between a lower bound $T_{\min}$ and an upper bound $T_{\max}$, the probability of transitioning to a session $p(t)$ follows a decaying exponential function, controlled by a temperature-like scaling factor $\tau$:

$$p(t) = \begin{cases} 1, & t < T_{\min} \\ \exp\left(-\frac{t - T_{\min}}{\tau}\right), & T_{\min} \le t < T_{\max} \\ 0, & t \ge T_{\max} \end{cases}.$$

In this formulation, $T_{\max}$ serves as a hard constraint to ensure that extremely long-horizon events (e.g., "a three-month project") are mandatory candidates for further decomposition, preventing the generation of unnaturally long or semantically overloaded single sessions.

#### B.3.3 LLM-Driven Adaptive Expansion

Beyond fixed temporal heuristics, KEME can leverage the latent reasoning capabilities of LLMs to make context-aware decisions. In this strategy, the knowledge-guided planner $\mathsf{A}_{\text{plan}}$ evaluates the semantic density and the complexity of the current event. This allows for an adaptive hierarchy where simple, routine tasks (e.g., "ordering a coffee") are realized as sessions immediately, while complex, multi-faceted activities (e.g., "planning a wedding") are expanded into deep sub-hierarchies regardless of their absolute duration or depth.

### B.4 Prompts Used in KEME

The prompts used in KEME consist of several components, including system-level prompts, task instructions, tool schemas and their runtime feedback, as well as data model schemas.

Given the large number and complexity of these prompts, we do not enumerate them exhaustively in this appendix. Instead, we refer interested readers to the released source code for complete prompt specifications and implementation details.

### B.5 Synthesis Settings

We implement KEME based on AgentScope Gao et al. (2025). The model used for trajectory synthesis is GPT-4.1 OpenAI (2025a), while GPT-5.2 OpenAI (2025b) is employed for question–answer pair generation and quality verification. During trajectory synthesis, we adopt a depth-based expansion strategy, where the maximum expansion depth $d_{\max}$ is set to 2. This means that when a node depth reaches 2, it can only be expanded into a session. For each temporal event graph, the minimum and maximum numbers of events, $m_{\min}$ and $m_{\max}$, are set to 1 and 15, respectively. The maximum number of tokens for the compatibility context of each event is set to 8,000. If this limit is exceeded, summarization based on GPT-4.1 is applied to reduce the token count. The temperature parameter is fixed at 1.0 throughout all stages.

**Algorithm 1** Closed-Loop Trajectory Synthesis in KEME

**Require:** Initial persona $\mathcal{P}_{\text{start}}$, trajectory's time horizon $[T_{\text{start}}, T_{\text{end}}]$, knowledge anchors $\mathcal{K}$, maximum depth $d_{\max}$, event-count bounds $(m_{\min}, m_{\max})$
**Ensure:** Synthesized trajectory $\mathcal{T}$, final persona $\mathcal{P}_{\text{end}}$

$r \leftarrow \mathcal{P}_{\text{start}}$ ▷ Set root node to the initial persona
$r.\text{start} \leftarrow T_{\text{start}}, \quad r.\text{end} \leftarrow T_{\text{end}}$ ▷ Each node has its start and end timestamps
$r.\mathcal{K} \leftarrow \mathcal{K}$ ▷ Each node has a set of corresponding knowledge anchors
$\mathcal{P} \leftarrow \mathcal{P}_{\text{start}}, \quad \mathcal{T} \leftarrow \emptyset$
$\mathcal{G}^{(r)} \leftarrow \textsc{ExpandNode}(r, \mathcal{P}, 1)$
$\mathcal{P}_{\text{end}} \leftarrow \mathcal{P}$
**return** $(\mathcal{T}, \mathcal{P}_{\text{end}})$
**function** $\textsc{ExpandNode}(u, \mathcal{P}, d)$
  $\mathcal{G}^{(u)} \leftarrow \mathsf{A}_{\text{plan}}\big(u, \mathcal{P}, d; m_{\min}, m_{\max}\big)$
  **for all** $k \in u.\mathcal{K}$ **do**
    $\mathcal{G}^{(u)} \leftarrow \mathsf{A}_{\text{ground}}\big(u, \mathcal{P}, \mathcal{G}^{(u)}, k; m_{\min}, m_{\max}\big)$ ▷ The knowledge anchor grounder may revise $\mathcal{G}^{(u)}$
  **end for**
  **while** $\textsc{ExistsUnexpandedNode}(\mathcal{G}^{(u)})$ **do** ▷ Enter expansion-refinement alternation
    $v \leftarrow \textsc{NextNodeInTopoOrder}(\mathcal{G}^{(u)})$
    **if** $d = d_{\max}$ **then**
      $(\mathcal{S}^{(v)}, \mathcal{P}) \leftarrow \mathsf{A}_{\text{realize}}\big(v, \mathcal{P}\big)$ ▷ The experience realizer may update persona
      Attach $\mathcal{S}^{(v)}$ as the output of $v$ and mark $v$ as expanded
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{\mathcal{S}^{(v)}\}$
      $o_v \leftarrow \mathcal{S}^{(v)}$
    **else**
      $\mathcal{G}^{(v)} \leftarrow \textsc{ExpandNode}(v, \mathcal{P}, d+1)$
      Attach $\mathcal{G}^{(v)}$ as the output of $v$ and mark $v$ as expanded
      $o_v \leftarrow \mathcal{G}^{(v)}$
    **end if**
    $\mathcal{G}^{(u)} \leftarrow \mathsf{A}_{\text{revise}}\big(u, \mathcal{P}, \mathcal{G}^{(u)}, v, o_v; m_{\min}, m_{\max}\big)$
  **end while**
  **return** $\mathcal{G}^{(u)}$
**end function**
**function** $\textsc{ExistsUnexpandedNode}(\mathcal{G})$
  **return** true iff $\mathcal{G}$ contains an event node whose output is not finalized
**end function**
**function** $\textsc{NextNodeInTopoOrder}(\mathcal{G})$
  **return** the next unexpanded node selected by a topological order induced by $\mathcal{G}$
**end function**

# C Additional Analysis

## C.1 Impact of Profile Schema Granularity on Trajectory Diversity

To study how profile schema complexity affects trajectory diversity, we define two reduced-dimension profile schemas by down-selecting dimensions from the original schema while preserving broad life-domain coverage. We refer to the original schema as a *fine-grained profile schema* (17 dimensions), and the two reduced variants as *medium-grained* (8 dimensions) and *coarse-grained* (6 dimensions). We randomly sample three persona seeds from PersonaHub Chan et al. (2024) and use GPT-4.1 OpenAI (2025a) to incrementally expand each seed into the fine-grained schema. Other schemas are then derived directly from the resulting fine-grained profile.

For trajectory synthesis, we set the maximum number of events to $m_{\max} = 10$ to reduce cost, and use an LLM-driven adaptive expansion strategy with a depth cap (event nodes at the maximum depth are forced to be realized as sessions), which typically produces shorter trajectories than non-LLM-driven alternatives. Following prior work Zhang et al. (2025c), we measure diversity on user

messages using lexical diversity (the distinct bigram ratio computed as the number of unique bigrams divided by the total number of bigrams) and semantic diversity ($1 - \hat{s}$, where $\hat{s}$ is the mean pairwise cosine similarity of user message embeddings computed by text-embedding-3-small[11]). To mitigate the effect of varying trajectory lengths, for each seed person we compute the minimum number of user messages among the three trajectories and sample that many user messages from each trajectory. We repeat sampling with five random seeds and report averaged results.

| Profile Schema | Lexical ↑ | Semantic ↑ |
|---|---|---|
| Coarse-grained | 0.7238 | 0.4783 |
| **Medium-grained** | **0.7313** | **0.4932** |
| Fine-grained | 0.7310 | 0.4875 |

Table 3: **Trajectory diversity under varying profile schema granularity.** Values are averaged across three seed persons and five length-matched sampling rounds.

Table 3 shows that the medium- and fine-grained schemas yield more diverse trajectories than the coarse-grained schema. However, the fine-grained schema does not provide a commensurate improvement over the reduced schemas. Leveraging KEME's evidence linking between messages and profile attributes, we observe that after synthesis only about $44\%$ of the fine-grained profile fields are ever mentioned, suggesting that many additional fields remain unused. We conjecture that this under-activation is primarily due to the limited trajectory length under our cost-motivated settings, which leaves insufficient opportunity for the generated experiences to surface and utilize the extra fine-grained details.

# D Discussion

## D.1 Comparison of Perception Approaches

Table 4 presents a comprehensive comparison between proposed protocol-based app-mediated perception and existing approaches, namely text-based perception and screenshot-based perception Tang et al. (2025). Most existing approaches implicitly assume that applications are passive observation targets, whose internal states must be inferred from screenshots or accessibility trees Wen et al. (2024); Li et al. (2024); Zhang et al. (2025a); Wang & Chen (2025); Volcengine (2025); Ye et al. (2025); Wu et al. (2025b). In contrast, our approach advocates for an app-mediated perception paradigm. This shift naturally leads to three consequences for memory systems:

**Significantly Reduced Redundancy.** User interactions within applications typically manifest as discrete, atomic actions, such as clicking a specific control or inputting a text string. In screenshot-based perception, the visual difference between frames captured immediately before and after such interactions is often negligible, leading to high *temporal redundancy*. Furthermore, screenshots inherently capture the entire User Interface (UI) state including static headers, advertisements, or background elements irrelevant to the current specific interaction. This introducs substantial *spatial noise*. Text-based parsing suffers from similar issues, often logging excessive structural details. In contrast, our protocol-based approach filters information directly at the source. **By decoupling the *sensing* logic from the *storing* logic, apps transmit only event-level summaries (e.g., "User completed a run") rather than raw interaction traces.** For memory systems, this dramatically lowers processing complexity, as they are relieved of the computational burden of inferring high-level user behaviors from low-level, noisy interaction trajectories. We note that this reduction relies on reasonable application-side summarization choices rather than representing an absolute bound.

**Enhanced Privacy and Safety Control.** Safety in memory systems necessitates strictly protecting user privacy while simultaneously respecting app boundaries. For certain applications, memory systems relying on screenshot-based or text-based parsing methods often risk violating Terms of Service (ToS), as they may function similarly to automated data crawlers. Furthermore, these methods typically capture on-screen information indiscriminately. **In contrast, our protocol-based approach allows both users and developers to collectively define the perception boundary.** It

[11] https://developers.openai.com/api/docs/models/text-embedding-3-small

| Perception Approach | Redundancy | Privacy & Safety Control | Processing Complexity | Interaction Mode |
|---|---|---|---|---|
| Text-Based Parsing | High | Low | High | Proactive |
| Screenshot-Based | High | Low | High | Proactive |
| **Protocol-Based App-Mediated** | **Low** | **High** | **Low** | **Passive** |

Table 4: **Comparison of perception interfaces for user-app interactions.** *Privacy & Safety Control* refers to the ability of both users and apps to define data boundaries. *Processing Complexity* denotes the computational burden placed on the memory system to extract valuable information from user interactions. *Interaction Mode* indicates whether the memory system proactively captures information or passively receives information from applications.

empowers users to explicitly select which applications participate in the memory formation process. Apps can sanitize data before transmission to ensure that sensitive content such as banking details or private chats remains unexposed. Consequently, protocol-based app-mediated perception achieves a higher standard of privacy compliance.

**Flexibility and Extensibility.** The protocol-based interface abstracts the heterogeneity of user interactions. **Instead of the memory system struggling to interpret diverse UI layouts across the dozens of applications typically installed on a user's device, each app implements its own lightweight perception module tailored to its content type.** As illustrated in Appendix A.2, a voice memo app can integrate a speech recognition module to send text, while a to-do list app uses rule-based triggers to report task creation or completion. This distributed perception capability allows the memory system to support heterogenous inputs seamlessly. Importantly, the shift to a passive interaction mode shifts the processing trigger from the memory system to the applications, significantly reducing the system's operational overhead. The memory system no longer acts as a bottleneck that must constantly monitor all active interfaces. It is noteworthy that protocol-based app-mediated perception complements existing perception methods and is most effective when applications can provide semantic signals at the cost of limited application-side support.